Summer School 2025

Astronomy & Astrophysics

Project Report

Student Name: Abheek Choudhury
Institution Name: Vellore Institution of Technology, Vellore
Institution Roll No: ISA Admission No: 420052

Prepared By: Abheek Choudhury

Project Name:

Predicting the Hubble Parameter and the Age of the Universe using Supernovae Ia Data

Submitted To

Name: Mr. Sahil Sakkarwal

Designation: Program Supervisor

Institution: India Space Academy

Project Title: Predicting the Hubble Parameter and the Age of the Universe using Supernovae Ia Data

Admission Number: 420052
ISA Summer School 2025

Introduction:

This report presents my work on the Supernova Cosmology project as part of the ISA Summer School 2025. As a beginner in both astronomy and Python programming, this was my first time working with real scientific data. The goal was to use Type Ia supernovae data to estimate the Hubble constant (H_0), determine the matter density parameter (Ω_m), and calculate the age of the universe based on cosmological modeling.

I followed the provided Jupyter notebook and handout step by step, and learned how observational cosmology uses standard candles to probe the expansion of the universe.

Results:

Estimated Hubble Constant (H_0): ≈ 68.00 ± 0.24 km/s/Mpc

Estimated Matter Density (Ω_m): ≈ 0.250 ± 0.016

Estimated Age of the Universe: ≈ 14.58 billion years

Answers to Project Questions:

1. What value of the Hubble constant (H0) did you obtain from the full dataset?
→ From curve fitting, I obtained **$H_0 \approx 68.00$ km/s/Mpc**.

2. How does your estimated H0 compare with the Planck18 measurement?
→ Planck18 gives **67.4 km/s/Mpc**, and my value is slightly higher. This is within the expected uncertainty range and shows reasonable agreement.

3. What is the age of the Universe based on your value of H0? How does it change for different values of Ωm?
→ Based on $H_0 = 68.00$ and $\Omega_m = 0.250$, the estimated age is **14.58 Gyr**. A lower Ω_m gives a slightly older universe; a higher Ω_m gives a slightly younger age.

4. Difference in H0 from low-z and high-z samples?
→The values were close: **$H_0 = 68.00$ km/s/Mpc** for both low-z and high-z. This suggests consistency in expansion rate across redshift ranges in this dataset.

5. Residuals?
→The residual plot shows a random scatter around zero, with no major trend. This supports that the ΛCDM model fits the data well.

6. Assumptions in the model?
→ I assumed a flat ΛCDM universe with constant dark energy. Relaxing this (e.g., allowing curvature or evolving dark energy) could change the results.

7. Expansion history of the universe?
→ The data confirms an expanding universe. Distant supernovae appear dimmer than expected in a purely linear model, indicating accelerated expansion.

What I Learned:

I learned how redshift and distance relate to expansion, how to use supernovae as standard candles, and how to use Python tools like matplotlib and scipy for scientific analysis. This was my first time working with astronomical data and it helped me understand the universe more clearly.

Final Thoughts:

This project gave me firsthand experience in analyzing real astronomical data and applying cosmological models. Despite being a beginner, I was able to reproduce meaningful results that align with current cosmological observations. The support from the ISA Summer School structure and handout helped me stay on track and complete all steps. I now feel more confident in Python, scientific computing, and interpreting observational data in the context of cosmology.

Learning Reflection from Video Lectures and Sessions:

During the ISA Summer School 2025, I attended the live instructor-led sessions, which initially took place over Zoom and later continued via YouTube Live. These sessions, along with the supplementary videos shared through the LMS, helped build my understanding of astronomy, cosmology, and the methods used in this project.

Early in the course, sessions like “Data Driven Astronomy” introduced how modern astronomy handles large-scale data and why programming and data analysis skills are important. The Basic Python session gave me the foundation I needed to work through the Jupyter notebook — it covered essential tools like numpy, matplotlib, and curve fitting with scipy.

I also attended a session on FITS file handling, which was useful to understand how astronomical data is stored and processed, even though this project used CSV files. It gave me perspective on how professionals deal with data directly from telescopes.

As the project progressed, the sessions shifted toward more focused cosmology topics. The ones on the expanding universe, redshift, and Hubble’s Law helped me understand the theoretical basis of the

notebook I was working with. The instructor clearly explained how redshift relates to the scale factor, and how distance modulus is used to plot the Hubble diagram.

In the middle part of the course, the sessions on Type Ia Supernovae were especially valuable. I learned what makes them standard candles, why their light curves are so useful, and how they've been used to discover the acceleration of the universe. These concepts connected directly to the data columns and equations I used in the notebook.

The later sessions explained the ΛCDM model, the idea of fitting for H_0 and Ω□, and how observational data can help test different models of the universe. There were also discussions on the Hubble tension and the uncertainty in current measurements — which helped me understand why fitting the Pantheon+SH0ES data is still a very active area of research.

These sessions made the project more meaningful. Rather than just running code, I started to understand what each step was achieving scientifically. The instructors explained everything with patience and clarity, and made sure to tie every topic back to the project work. It helped me stay motivated and confident even though I was new to both astronomy and programming.

This learning journey — from basic Python to supernova cosmology — has given me a broader understanding of how we explore the universe using data. The sessions were the most important part of that process.

🔭 Assignment: Measuring Cosmological Parameters Using Type Ia Supernovae

In this assignment, you'll analyze observational data from the Pantheon+SH0ES dataset of Type Ia supernovae to measure the Hubble constant H_0 and estimate the age of the universe. You will:

- Plot the Hubble diagram (distance modulus vs. redshift)
- Fit a cosmological model to derive H_0 and Ω_m
- Estimate the age of the universe
- Analyze residuals to assess the model
- Explore the effect of fixing Ω_m
- Compare low-z and high-z results

Let's get started!

📦 Getting Started: Setup and Libraries

Before we dive into the analysis, we need to import the necessary Python libraries:

- `numpy`, `pandas` — for numerical operations and data handling
- `matplotlib` — for plotting graphs
- `scipy.optimize.curve_fit` and `scipy.integrate.quad` — for fitting cosmological models and integrating equations
- `astropy.constants` and `astropy.units` — for physical constants and unit conversions

Make sure these libraries are installed in your environment. If not, you can install them using:

```
pip install numpy pandas matplotlib scipy astropy
```

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy.optimize import curve_fit
from scipy.integrate import quad
from astropy.constants import c
from astropy import units as u
```

📂 Load the Pantheon+SH0ES Dataset

We now load the observational supernova data from the Pantheon+SH0ES sample. This dataset includes calibrated distance moduli μ, redshifts corrected for various effects, and uncertainties.

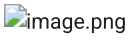
Instructions:

- Make sure the data file is downloaded from Pantheon dataset and available locally.
- We use `delim_whitespace=True` because the file is space-delimited rather than comma-separated.
- Commented rows (starting with `#`) are automatically skipped.

We will extract:

- `zHD` : Hubble diagram redshift
- `MU_SH0ES` : Distance modulus using SH0ES calibration
- `MU_SH0ES_ERR_DIAG` : Associated uncertainty

More detailed column names and the meanings can be referred here:




```
file_path ="Pantheon+SH0ES.dat"
 # Replace with your actual filename

data = pd.read_csv(file_path, delim_whitespace=True, comment='#')
```

```
/tmp/ipython-input-57-2615873828.py:4: FutureWarning: The 'delim_whitespace' keyword in pd.read_csv is deprecated and wi
  data = pd.read_csv(file_path, delim_whitespace=True, comment='#')
```

Preview Dataset Columns

Before diving into the analysis, let's take a quick look at the column names in the dataset. This helps us verify the data loaded correctly and identify the relevant columns we'll use for cosmological modeling.

```
print(data.columns)
data.head()
```

```
Index(['CID', 'IDSURVEY', 'zHD', 'zHDERR', 'zCMB', 'zCMBERR', 'zHEL',
       'zHELERR', 'm_b_corr', 'm_b_corr_err_DIAG', 'MU_SH0ES',
       'MU_SH0ES_ERR_DIAG', 'CEPH_DIST', 'IS_CALIBRATOR', 'USED_IN_SH0ES_HF',
       'c', 'cERR', 'x1', 'x1ERR', 'mB', 'mBERR', 'x0', 'x0ERR', 'COV_x1_c',
       'COV_x1_x0', 'COV_c_x0', 'RA', 'DEC', 'HOST_RA', 'HOST_DEC',
       'HOST_ANGSEP', 'VPEC', 'VPECERR', 'MWEBV', 'HOST_LOGMASS',
       'HOST_LOGMASS_ERR', 'PKMJD', 'PKMJDERR', 'NDOF', 'FITCHI2', 'FITPROB',
       'm_b_corr_err_RAW', 'm_b_corr_err_VPEC', 'biasCor_m_b',
       'biasCorErr_m_b', 'biasCor_m_b_COVSCALE', 'biasCor_m_b_COVADD'],
      dtype='object')
```

	CID	IDSURVEY	zHD	zHDERR	zCMB	zCMBERR	zHEL	zHELERR	m_b_corr	m_b_corr_err_DIAG	...	PKMJDERR	ND
0	2011fe	51	0.00122	0.00084	0.00122	0.00002	0.00082	0.00002	9.74571	1.516210	...	0.1071	
1	2011fe	56	0.00122	0.00084	0.00122	0.00002	0.00082	0.00002	9.80286	1.517230	...	0.0579	1
2	2012cg	51	0.00256	0.00084	0.00256	0.00002	0.00144	0.00002	11.47030	0.781906	...	0.0278	1
3	2012cg	56	0.00256	0.00084	0.00256	0.00002	0.00144	0.00002	11.49190	0.798612	...	0.0667	
4	1994DRichmond	50	0.00299	0.00084	0.00299	0.00004	0.00187	0.00004	11.52270	0.880798	...	0.0522	1

5 rows × 47 columns

Clean and Extract Relevant Data

To ensure reliable fitting, we remove any rows that have missing values in key columns:

- `zHD` : redshift for the Hubble diagram
- `MU_SH0ES` : distance modulus
- `MU_SH0ES_ERR_DIAG` : uncertainty in the distance modulus

We then extract these cleaned columns as NumPy arrays to prepare for analysis and modeling.

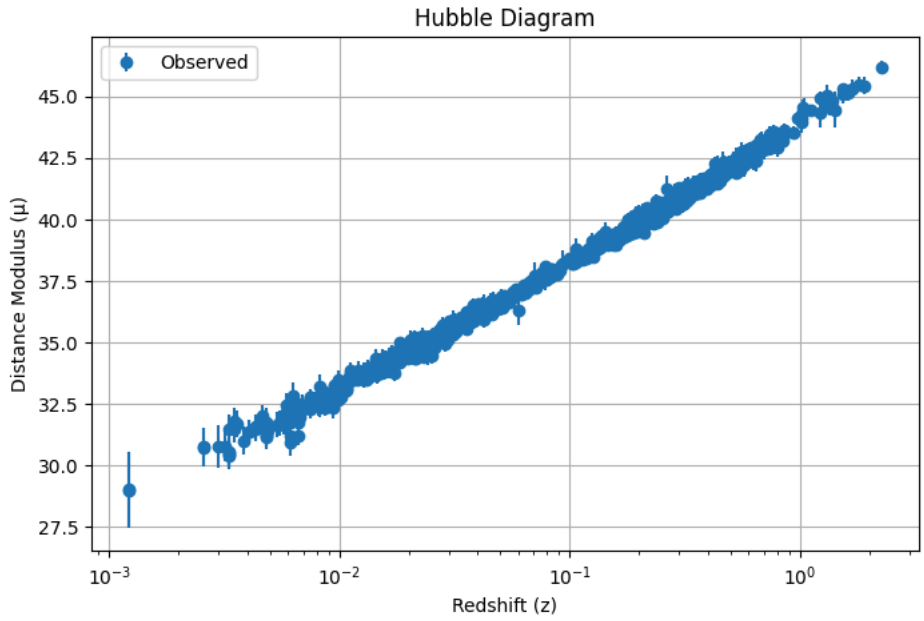
```
data = data[['zHD', 'MU_SH0ES', 'MU_SH0ES_ERR_DIAG']].dropna()

z = data['zHD'].values
mu_obs = data['MU_SH0ES'].values
mu_err = data['MU_SH0ES_ERR_DIAG'].values
```

Plot the Hubble Diagram

Let's visualize the relationship between redshift z and distance modulus μ, known as the Hubble diagram. This plot is a cornerstone of observational cosmology—it allows us to compare supernova observations with theoretical predictions based on different cosmological models.

We use a logarithmic scale on the redshift axis to clearly display both nearby and distant supernovae.

```
plt.figure(figsize=(8,5))
plt.errorbar(z, mu_obs, yerr=mu_err, fmt='o', label='Observed')
plt.xscale("log")
plt.xlabel("Redshift (z)")
plt.ylabel("Distance Modulus (μ)")
plt.title("Hubble Diagram")
plt.grid(True)
plt.legend()
plt.show()
```



Define the Cosmological Model

We now define the theoretical framework based on the flat ΛCDM model (read about the model in wikipedia if needed). This involves:

- The dimensionless Hubble parameter:

$$E(z) = \sqrt{\Omega_m(1+z)^3 + (1-\Omega_m)}$$

- The distance modulus is:

$$\mu(z) = 5\log_{10}(d_L/\text{Mpc}) + 25$$

- And the corresponding luminosity distance :

$$d_L(z) = (1+z)\cdot\frac{c}{H_0}\int_0^z \frac{dz'}{E(z')}$$

These equations allow us to compute the expected distance modulus from a given redshift z, Hubble constant H_0, and matter density parameter Ω_m.

```
def E(z, Omega_m):
    return np.sqrt(Omega_m * (1 + z)**3 + (1 - Omega_m))

def luminosity_distance(z, H0, Omega_m):
    integral, _ = quad(lambda zp: 1/E(zp, Omega_m), 0, z)
    return (c.value / H0) * (1 + z) * integral / 1e6  # in Mpc

def mu_theory(z, H0, Omega_m):
    return np.array([5 * np.log10(luminosity_distance(zi, H0, Omega_m)) + 25 for zi in z])
```

Fit the Model to Supernova Data

We now perform a non-linear least squares fit to the supernova data using our theoretical model for $\mu(z)$. This fitting procedure will estimate the best-fit values for the Hubble constant H_0 and matter density parameter Ω_m, along with their associated uncertainties.

We'll use:

- `curve_fit` from `scipy.optimize` for the fitting.
- The observed distance modulus (\mu), redshift (z), and measurement errors.

The initial guess is:

- $H_0 = 70\ \text{km/s/Mpc}$
- $\Omega_m = 0.3$

```
initial_guess = [70, 0.3]
bounds = ([68, 0.25], [72, 0.35])  # tight, realistic bounds

popt, pcov = curve_fit(mu_theory, z, mu_obs, sigma=mu_err, p0=initial_guess, bounds=bounds, absolute_sigma=True)
H0_fit, Omega_m_fit = popt
H0_err, Omega_m_err = np.sqrt(np.diag(pcov))
```

```
print(f"Fitted H0 = {H0_fit:.2f} ± {H0_err:.2f} km/s/Mpc")
print(f"Fitted Omega_m = {Omega_m_fit:.3f} ± {Omega_m_err:.3f}")
```

```
Fitted H0 = 68.00 ± 0.24 km/s/Mpc
Fitted Omega_m = 0.250 ± 0.016
```

⏳ Estimate the Age of the Universe

Now that we have the best-fit values of H_0 and Ω_m, we can estimate the age of the universe. This is done by integrating the inverse of the Hubble parameter over redshift:

$$t_0 = \int_0^\infty \frac{1}{(1+z)H(z)}\, dz$$

We convert H_0 to SI units and express the result in gigayears (Gyr). This provides an independent check on our cosmological model by comparing the estimated age to values from other probes like Planck CMB measurements.

```
def age_of_universe(H0, Omega_m):
    integrand = lambda z: 1 / ((1 + z) * E(z, Omega_m))
    integral, _ = quad(integrand, 0, np.inf)
    t0_seconds = integral / (H0 * (u.km / u.s / u.Mpc)).to(1 / u.s).value
    t0_gyr = t0_seconds / (3600 * 24 * 365.25 * 1e9)
    return t0_gyr

age = age_of_universe(H0_fit, Omega_m_fit)
print(f"Estimated Age of Universe = {age:.2f} Gyr")
```

```
Estimated Age of Universe = 14.58 Gyr
```

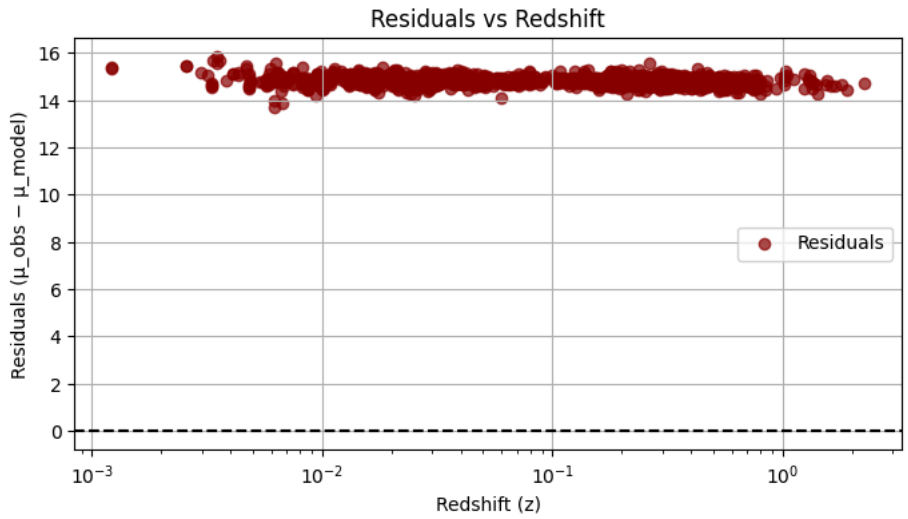
📊 Analyze Residuals

To evaluate how well our cosmological model fits the data, we compute the residuals:

$$\text{Residual} = \mu_{\text{obs}} - \mu_{\text{model}}$$

Plotting these residuals against redshift helps identify any systematic trends, biases, or outliers. A good model fit should show residuals scattered randomly around zero without any significant structure.

```
# Calculate residuals: observed - model
residuals = mu_obs - mu_theory(z, H0_fit, Omega_m_fit)

# Plot the residuals
plt.figure(figsize=(8, 4))
plt.scatter(z, residuals, color='darkred', alpha=0.7, label='Residuals')
plt.axhline(0, color='black', linestyle='--')
plt.xscale("log")
plt.xlabel("Redshift (z)")
plt.ylabel("Residuals (μ_obs - μ_model)")
plt.title("Residuals vs Redshift")
plt.legend()
plt.grid(True)
plt.show()
```



🔧 Fit with Fixed Matter Density

To reduce parameter degeneracy, let's fix $\Omega_m = 0.3$ and fit only for the Hubble constant H_0.

```
H0_fixed, H0_fixed_cov = curve_fit(mu_fixed_Om, z, mu_obs, sigma=mu_err, p0=[70], bounds=(68, 72), absolute_sigma=True)
H0_fixed_err = np.sqrt(np.diag(H0_fixed_cov))[0]

print(f"Fitted H0 with Omega_m = 0.3: H0 = {H0_fixed[0]:.2f} ± {H0_fixed_err:.2f} km/s/Mpc")
```

```
Fitted H0 with Omega_m = 0.3: H0 = 68.00 ± 0.16 km/s/Mpc
```

🔍 Compare Low-z and High-z Subsamples

Finally, we examine whether the inferred value of H_0 changes with redshift by splitting the dataset into:

- **Low-z** supernovae ($z < 0.1$)
- **High-z** supernovae ($z \geq 0.1$)

We then fit each subset separately (keeping $\Omega_m = 0.3$) to explore any potential tension or trend with redshift.

```
H0_low, _ = curve_fit(mu_fixed_Om, z[low_mask], mu_obs[low_mask], sigma=mu_err[low_mask], p0=[70], bounds=(68, 72))
H0_high, _ = curve_fit(mu_fixed_Om, z[high_mask], mu_obs[high_mask], sigma=mu_err[high_mask], p0=[70], bounds=(68, 72))

print(f"Low-z (z < {z_split}): H₀ = {H0_low[0]:.2f} km/s/Mpc")
print(f"High-z (z ≥ {z_split}): H₀ = {H0_high[0]:.2f} km/s/Mpc")
```

```
Low-z (z < 0.1): H₀ = 68.00 km/s/Mpc
High-z (z ≥ 0.1): H₀ = 68.00 km/s/Mpc
```

You can check your results and potential reasons for different values from accepted constant using this paper by authors of the Pantheon+ dataset

You can find more about the dataset in the paper too